

Symbolischer Prozessor

Computer für die speziellen Probleme der Künstlichen Intelligenz sind meistens sehr komplex und teuer. Der Ivory-Chip könnte diese Situation demnächst gründlich ändern.

Vor kurzen stellte der amerikanische Hersteller Symbolics einen Mikroprozessor vor, der ungewöhnlich schnell Lisp-Programme ausführt. Bekanntlich lassen sich mit dieser Hochsprache die typischen Probleme im Bereich der Künstlichen Intelligenz (KI) gut lösen. Aber Programmierer, die mit der Hochsprache Lisp allgemein formulierte mathematische Aufgaben bewältigen wollen, benötigen sehr leistungsfähige Computer. Und genau diese Forderung erfüllt der rasante „Ivory-Chip“ S40959 mit seiner neuartigen Architektur: Der Entwicklung großer und ehrgeiziger Software-Projekte steht durch diesen Mi-

intosh II von Apple oder einen IBM-kompatiblen Rechner stecken. Somit eröffnen sich für den Einsatz der Künstlichen Intelligenz und der Symbolverarbeitung neue Perspektiven: Beide können jetzt auf breiter Basis dem „Normalverbraucher“ zugänglich gemacht werden. Im übrigen sollen nicht nur Lisp, sondern auch Compiler für die weitverbreiteten Hochsprachen Prolog, Ada, Pascal und Fortran demnächst verfügbar sein.

Dem Systemprogrammierer bietet der neue Mikroprozessor von Symbolics viele interessante Möglichkeiten, die Programmentwicklung drastisch zu vereinfachen und zu verkürzen. Schon auf unterster Ebene, also wenn der Ivory-Prozessor sein Maschinensprache-Programm ausführt, testet die Elektronik auf dem Schaltkreis die Verträglichkeit der verwendeten Datentypen und überprüft automatisch, ob beispielsweise ein Array die erlaubten Grenzen überschreitet. Mithin kann der Mikroprozessor bereits auf Maschinenebene undefinierte Variablen und Funktionen selbständig

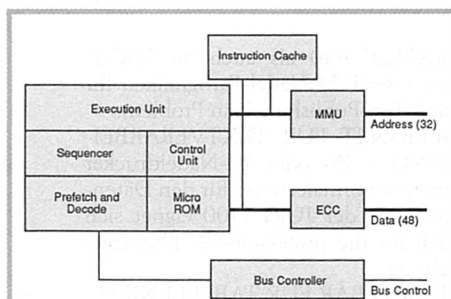
erkennen. Und auf Wunsch prüft der Schaltkreis auch die Anzahl der mitgegebenen Variablen bei einem beliebigen Funktionsaufruf. Solche Testaufgaben können selbst die stärksten klassischen Mikroprozessoren wie der 68030 von Motorola oder der 80386 von Intel nur mit relativ zeitaufwendigen Maschinenprogrammen lösen. Die Hardware des Ivory-Prozessors erledigt derartige Probleme wesentlich schneller, als dies durch Software möglich wäre.

Überhaupt verfügt der Ivory-Chip über einen Befehlssatz, der modulares Programmieren komplexer Software-Projekte geradezu herausfordert: Sogar dann, wenn der Prozessor ein neues, zu testendes Programm ausführt, kann der Software-Designer bestimmte Programmteile modifizieren. Die langen Compilierzeiten großer Software-Projekte reduzieren sich durch den Ivory-Prozessor erheblich, da der Software-Entwickler nur die fehlerhaften Programmteile neu compiliert und sie anschließend in das Gesamtprogramm einbindet.

IVORY-CHIP

kroprozessor kaum etwas im Wege.

Sicherlich beschritten die Entwickler völlig neue Bahnen, als sie den Ivory-Chip entwarfen. Seine Architektur ist weit vorausschauend angelegt, so daß der Mikroprozessor KI-Anwendungen mit Hilfe der Parallelverarbeitung vortrefflich meistern kann. Sogar der Hardware-Aufwand hält sich in Grenzen: Die verfügbaren KI-Computer bestehen gewöhnlich aus mehreren komplexen Steckkarten. Im Gegensatz dazu läßt sich ein Computer, der auf dem Ivory-Prozessor basiert, samt Arbeitsspeicher und Interface-Bausteinen leicht auf einer einzigen Platine unterbringen. Eine entsprechende Karte kann der Anwender dann beispielsweise in den Mac-



Das Block-Schaltbild zeigt die einzelnen Baugruppen des Ivory-Prozessors. Mit der Fehlerkorrektur-Logik (ECC) repariert der Chip auftretende Bus-Fehler selbständig.

Betrachtet man das Block-Schaltbild des Ivory-Chips, so wird schnell sichtbar, was in ihm steckt. Der Mikroprozessor adressiert den Arbeitsspeicher als vier voneinander unabhängige Speicherbänke. Welche Bank adressiert wird, entscheiden die niederen zwei Adreß-Bits des 32 bit breiten Adreß-Busses. Dieses Konzept führt dazu, daß der Schaltkreis die einzelnen Bänke verschachtelt adressiert – dadurch steigert sich der Durchsatz auf dem Daten-Bus im Vergleich zum klassischen Mikroprozessor beträchtlich. Im Prinzip versucht der Ivory-Prozessor bei der Ausführung eines Befehls stets solche Speicherbänke anzusprechen, die während des momentanen Taktzyklus

*Aufgrund der hohen Integrationsdichte
paßt ein nahezu kompletter KI-(Künstliche
Intelligenz)-Computer auf ein
einziges Siliziumplättchen*

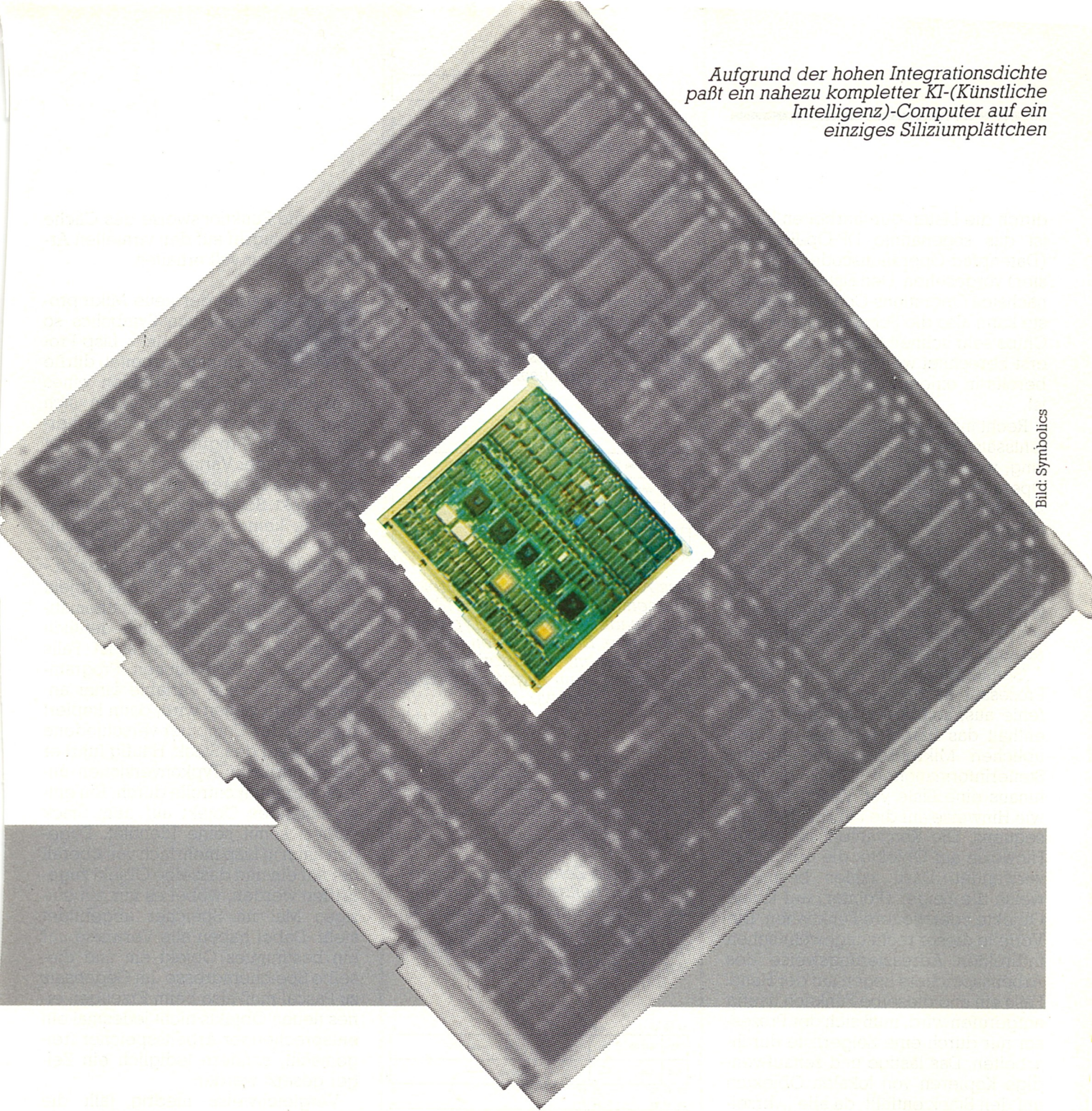


Bild: Symbolics

nicht damit beschäftigt sind, Daten zu übernehmen oder zum Mikroprozessor zu senden. Nur wenn alle vier Speicherbänke gleichzeitig beschäftigt sind, muß der Ivory-Chip kurzzeitig warten. Neben dem eigentlichen Adreß-Bus enthält der Schaltkreis noch einen 10-bit-Interface-Bus für den direkten Anschluß eines entsprechenden Arithmetikprozessors.

Um auch Bus-Fehler selbständig reparieren zu können, ist der Ivory-Prozessor mit einem 48 bit breiten Daten-Bus ausgestattet. Zwar arbeitet der Schaltkreis nur mit 32 bit breiten Daten, wie zum Beispiel auch der 68030 von Motorola oder der 80386 von Intel. Allein acht zusätzliche Daten-Bit benötigt die Fehlererkennungs- und Kor-

rekturlogik auf dem Halbleiterplättchen. Einzelne und doppelte Bit-Fehler innerhalb eines 32-bit-Wortes sowie Mehrfachfehler in einem sogenannten Daten-Nibble (4 bit) kann der Lisp-Prozessor ohne Probleme selbständig korrigieren. Somit bietet schon die Hardware des Mikroprozessors ungewöhnlich viele Sicherheitsreserven in einem professionellen Computer-System.

Bei der momentanen Version arbeitet der Ivory-Prozessor mit 32 internen und 64 externen Maschinenregistern. Wie der Name schon sagt, sind die internen Register auf dem Prozessor-Chip integriert, und die externen befinden

sich im Arbeitsspeicher. Darüber hinaus verfügen die meisten Register über eine festverdrahtete Datentypen-Logik: In ein solches Register kann beispielsweise ein Operand mit einem bestimmten Datentyp geschrieben werden, aber nach einer arithmetischen Operation enthält das gleiche Register einen ganz anderen typverschiedenen Operanden. Mit anderen Worten: Der Schaltkreis führt bei Bedarf eine Typenkonversion automatisch durch – ein Problem, das herkömmliche Mikroprozessoren nur durch zeitaufwendige Routinen meistern.

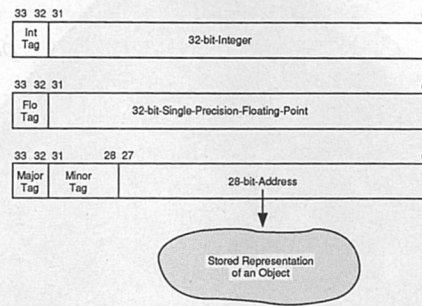
Bekanntlich ist Lisp eine listenorientierte Programmiersprache. Damit sich der Ivory-Prozessor einfach

durch die Listen durcharbeiten kann, ist das sogenannte DP-Op-Register (Datenpfad-Operationscode-Register) vorgesehen. Den Datenpfad zum nächsten Operations-Code in einer Liste kann also die Schaltungslogik des Chips sehr schnell finden, da er nicht erst berechnet werden muß, sondern bereits in einem Register vorhanden ist.

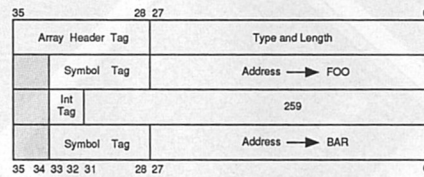
Recht interessant sieht auch der Befehlssatz aus: Jede Instruktion ist 17 bit lang, wovon neun bit der eigentliche Operations-Code und acht bit das dazugehörige Argument einnehmen. Somit lassen sich zwei Befehle in ein einziges Datenwort packen. Im Vergleich zu den herkömmlichen Mikroprozessoren fällt der Maschinen-Code des Ivory-Chips nicht nur kompakter aus, sondern er läßt sich auch wesentlich rasanter ausführen.

Genau genommen führt der Ivory-Prozessor auch keine Maschinenbefehle aus, sondern Funktionen. Zwar enthält das Befehlswort wie bei den üblichen Mikroprozessoren interne Steuerinformationen, aber darüber hinaus eine Liste von Konstanten sowie Hinweise auf die nächste Befehlssequenz. Die Konstantenliste enthält Hinweise auf Objekte, die der Befehl verwendet. Dazu zählen beispielsweise die Zeiger (Pointer) auf lokale Objekte oder weitere Funktionen. Die Vorteile dieser recht ungewöhnlichen indirekten Adressierungsweise des Arbeitsspeichers liegen auf der Hand: Falls ein und dieselbe Funktion häufig aufgerufen wird, muß sich der Prozessor nur durch eine Zeigerliste durcharbeiten. Das lästige und zeitaufwendige Kopieren von lokalen Objekten auf den Stack entfällt, da alle „verzeigerten“ Funktionen auf einen gemeinsamen „Daten-Pool“ zugreifen. Selbst rekursive (sich selbst aufrufende) Funktionen lassen sich somit ungewöhnlich einfach auf Maschinenebene verwirklichen.

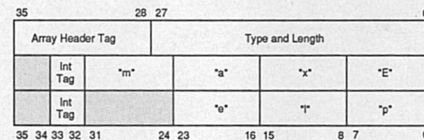
Um jede Art von Befehlen besonders effektiv auszuführen, ist der Ivory-Prozessor mit einem Befehls-Cache ausgerüstet. Es handelt sich dabei um einen schnellen Zwischenspeicher, in den der Prozessor den vom Arbeitsspeicher ankommenden Datenstrom temporär ablegt. Deshalb sind beispielsweise bei kleineren Programmschleifen keine externen Zugriffe auf den vergleichsweise trägen Arbeitsspeicher mehr nötig. Selbstverständlich



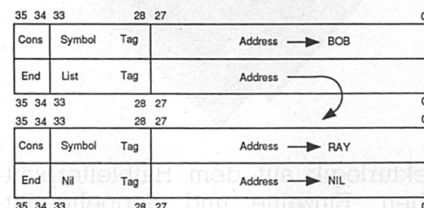
Ein Objekt setzt sich entweder aus einem 32-bit-Datenwort mit einem 2-bit-Datentyp-Feld oder einer 28-bit-Adresse mit einem 6-bit-Datenfeld zusammen



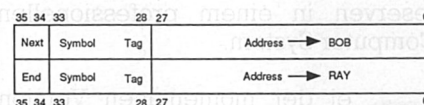
Das Array besteht aus den drei Elementen FOO, 259 und BAR. Der Header enthält den Typ und die Länge. Danach folgen für jedes Element die Referenzen. Sie geben beim Zugriff Aufschluß über den Datentyp.



So wird die Zeichenkette „Example“ im Arbeitsspeicher abgebildet. Jedes Zeichen wird dabei in ein Byte (8 bit) gepackt. Intern arbeitet jedoch der Mikroprozessor nur mit 32-bit-Objekten.



Eine Liste, bestehend aus zwei Elementen, nimmt nur vier Worte ein. Im Gegensatz zum Array hat eine Liste keinen „Kopf“ (Header).



Gepackte Listen nehmen zwei Worte ein. Um Referenzen auf andere Objekte zu eliminieren, verwendet der Prozessor einen speziellen Code.

bleibt die Funktionsweise des Cache auch bei Zugriff auf den virtuellen Arbeitsspeicher voll erhalten.

Warum der neue Mikroprozessor von Symbolics so schnell ein Lisp-Programm verarbeitet, dürfte nun klar sein: Die Variablen eines Lisp-Programms unterscheiden sich grundsätzlich von denen eines herkömmlichen Programms. In Lisp ist der Wert einer Variablen nichts anderes als die Referenz auf ein bestimmtes Objekt. Somit haben die einzelnen Objekte keine so strenge Typenbindung wie beispielsweise in Pascal. Vielmehr ist in Lisp der Typ eines Objekts in der Referenz auf das Objekt codiert. Anhand einer mathematischen Zuweisung läßt sich deutlich zeigen, was damit gemeint ist: Falls mit einer herkömmlichen Programmiersprache eine Variable einer anderen zugewiesen wird, dann kopiert zunächst der Computer verschiedene Objekte auf den Stack. Häufig führt er dabei implizite Typkonversionen unter Programmkontrolle durch. Ein entsprechendes Objekt auf dem Stack verliert somit seine Identität. Dagegen kann in Lisp mehrfach von überall her auf ein und dasselbe Objekt zugegriffen werden, wobei es nur ein einziges Mal im Speicher abgebildet steht. Dabei haben alle Verweise auf ein bestimmtes Objekt ein und dieselbe Speicheradresse. Im Gegensatz zu Pascal muß also beim Erzeugen eines neuen Objekts nicht jedesmal ein entsprechender Arbeitsspeicher freigemacht, sondern lediglich ein Zeiger gesetzt werden.

Vergleichsweise niedrig fällt die Taktfrequenz des Ivory-Prozessors aus: Sie beträgt nur acht Megahertz. Aber wegen des verschachtelten Zugriffs auf den in vier Bänke aufgeteilten Arbeitsspeicher läßt sich dennoch ein hoher Durchsatz auf dem Datenbus erzielen. Und beim Wiederauffrischen von dynamischen RAM-Schaltkreisen muß meistens der Mikroprozessor wegen der niedrigen Taktrate erst gar nicht angehalten werden (Hidden Refresh).

Momentan läuft der Mikroprozessor nur in einer KI-Workstation von Symbolics; jedoch sollen demnächst Erweiterungskarten mit dem neuen Mikroprozessor für den Macintosh II von Apple, für AT-kompatible Computer, für IBM PS/2 und die Sun-Workstation verfügbar sein.

-na